MASTER THESIS

TITLE: Achieving Proportional Fairness in WiFi Networks via Convex Bandit Optimization
MASTER DEGREE: Master's degree in Applied Telecommunications and Engineering Management (MASTEAM)
AUTHOR: Golshan Famitafreshi
ADVISOR: Cristina Cano
TUTOR: David Rincón
DATE: October, 23rd 2018

Title: Achieving Proportional Fairness in WiFi Networks via Convex Bandit Optimization

Author: Golshan Famitafreshi

Advisor: Cristina Cano

Tutor: David Rincón

Date: October 23, 2018

Abstract

In the last years proportional fairness has attracted attention in the literature on multi-rate IEEE 802.11 WLANs. One way to improve the performance of wireless networks is contention window tuning based on proportional fairness. In this thesis, we investigate how to apply a bandit convex optimization algorithm - a powerful framework for wireless network optimization - to proportional fair resource allocation in wireless networks. We propose an algorithm which is able to learn the optimal slot transmission probability only by monitoring the throughput of the network. We have evaluated the Online Gradient Descent with Sequential Multi-Point Gradient Estimates algorithm both by using the true value of the function to optimize, as well as adding estimation errors by using a network simulator. By means of the proposed algorithm, we provide extensive experimental results which illustrate the sensitivity of the algorithm to different exploration schedules, exploration parameters and gradient descent step size. We also show the sensitivity of the algorithm to noisy gradient estimates. We believe this research can be considered as a practical solution in order to improve the performance of wireless networks, in particular, in commercial WiFi cards.

ACKNOWLEDGMENTS

I would like to acknowledge and give special thanks to my thesis advisor Dr. Cristina Cano at Universitat Oberta de Catalunya for her valuable guidance, encouragement, and patience in the entire work.

I would also like to thank my thesis tutor Dr.David Rincón at Universitat Politecnica de Catalunya BarcelonaTech for his valuable comments, reviewing and his contribution in my work.

This work is partially supported by the Wireless Networks Research lab at Universitat Oberta de Catalunya.

Finally, I would like to dedicate this research to my parents and to my sister, who encouraged me in every step. Your efforts and support made me strong to follow up a higher education. Your patience and love made this research possible. Thank you for your unlimited love.

Contents

INTRODUCTION **1**

1 IEEE 802.11 BACKGROUND **2**

1.1 IEEE 802.11 MAC 2

1.2 Throughput Optimization in WiFi Networks 3

2 WiFi PROPORTIONAL FAIRNESS VIA BANDIT CONVEX OPTIMIZATION **6**

2.1 Bandit Convex Optimization 6

2.2 Sequential Multi-Point Gradient Estimates in WiFi 7

3 PERFORMANCE EVALUATION **11**

3.1 Methodology 11

3.2 Simulation Results 14

3.2.1 Evaluation of Convergence without Noise 15

3.2.1.1 Sensitivity to the Learning Parameters 15

3.2.1.2 Sensitivity to the Exploration Schedules 16

3.2.2 Evaluation of Convergence Adding Noise 18

3.2.2.1 Sensitivity to Noisy Gradient Estimates 21

3.2.2.2 Sensitivity to Different Temporal Batches 23

4 RELATED WORK **25**

5 CONCLUSION **27**

5.1 Contributions 27

5.2 Future Work 28

5.3 Sustainability Consideration 29

5.4 Ethical Consideration 29

ACRONYMS **30**

Bibliography **31**

INTRODUCTION

Bandit Convex Optimization is a type of Online Convex Optimization (OCO) in which we deal with partial information. In BCO, decisions are made between a player and an adversary repeatedly. In each iteration, the player selects a point from a fixed and known convex set. Then the adversary chooses a convex cost function. At the end of the iteration, the only available feedback for the player is the cost of the function at the selected point. In this framework, the player does not have any knowledge about the specific function nor the gradient [1]. The main emphasis of BCO in the machine learning community has been on rigorous theoretical performance analysis of algorithms. However, practical application of BCO algorithms still require more attention.

We argue that since many wireless network optimization problems can be easily formulated as convex problems; BCO is appealing for the wireless networking community. Some potential applications of the convex optimization formulation are pulse shaping filter design [2], transmit beamforming [3], network resource allocation [4], MMSE precoder design for multi-access communication [5], robust beamforming [6] and optimal linear decentralized estimation [7]. Therefore, by using BCO in the optimization of wireless communications it is easy to reformulate these problems into convex problems with partial information. BCO has two important advantages in wireless network communication. The first advantage is that the player only needs the cost function feedback of a given action, which facilitates practical implementation. Second, in BCO, the adversary is able to choose among a set of convex functions which can capture network dynamics such as changes in the number of nodes and channel conditions.

In this thesis, we investigate how to apply a bandit convex optimization algorithm to proportional fair resource allocation in wireless networks. This approach can be implemented by the access point allowing learning of the optimal slot transmission probability only by monitoring the throughput of the network. This research can help academia and the industry to assess whether bandit convex optimization algorithms can be a practical solution for commercial WiFi cards.

This Master thesis is organized as follows: Section 1 describes the main background: the IEEE 802.11 MAC operation and throughput optimization in WiFi networks. Section 2 explains WiFi proportional fairness via bandit convex optimization and our proposed algorithm. Section 3 describes the methodology of this project which includes a description of the simulator and its parameters. Also, in this section, the evaluation of the method and the results obtained are presented. Section 4 summarizes the related work in the area of wireless network throughput optimization and the benefits of applying BCO to this problem. In Section 5 some final remarks are given. Finally, in Section 6 all the expansions of abbreviations are expressed.

CHAPTER 1. IEEE 802.11 BACKGROUND

This research is done based on three main concepts which are briefly explained in this section and the following one. In this section first, we explain the IEEE 802.11 DCF mechanism, then summarize the throughput optimization in WiFi networks based on proportional fairness.

1.1 IEEE 802.11 MAC

The IEEE 802.11 protocol consists of physical layer and media access control specifications for WLAN communications in various frequency bands ranging from 900 MHz to 60 GHz. Protocols which define the regulations for WiFi are based on the standards and amendments versions. The fundamental mechanism of the 802.11 protocol is known as the Distributed Coordination Function (DCF). It employs a Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA) method with binary exponential backoff. The DCF default access technique is a two-way handshaking scheme called basic access mechanism, which consists of sending a data packet which is replied by an acknowledgment.

In DCF when a station wants to send a packet, it monitors the channel. If it senses the channel idle for a specific interval it will start a back-off countdown. This time interval is known as distributed interframe space (DIFS). Otherwise, if a station senses the channel busy, it continues to monitor the channel until it is sensed idle for a DIFS. After the channel is sensed idle for a DIFS, the back-off countdown timer starts. DCF operates in a discrete-time back-off scale. The time after each DIFS is slotted and each station is able to start a transmission only at the beginning of each slot time. The duration of this time slot (σ) is specified in the standard. This value accounts for the propagation delay and can be considered as one of the physical layer parameters. Each time a station starts the back-off procedure it initializes CW to $\mathrm{CW}_{\min}$ and chooses a random number in $(0, \mathrm{CW}-1)$, where CW is the contention window. The back-off timer is then computed as $\mathrm{CW}\sigma$ and as long as the channel is sensed idle for a time slot the back-off timer counter decrements. When a transmission is detected on the channel, this timer freezes. It is reactivated when the channel is sensed idle for more than DIFS. After each failed transmission (either due to channel errors or collisions with other simultaneous transmissions) the CW is doubled. The maximum value for contention window is equal to $2^{\mathrm{m}}\mathrm{CW}_{\min}$. The value m is the maximum back-off stage and it is a configurable parameter. Once a packet is transmitted the sender waits for an ACK confirming the correct reception of the packet. If the station that started the transmission does not receive an ACK during the ACK timeout period, it understands that a collision happened.[1] Therefore, the station retransmits the packet according to the back-off process. If a packet experiences more collisions than the maximum retry limit, the packet will be discarded.

[1] Note that this can also occur due to channel errors.

In Fig. 1.1 we show two stations (A and B) that share a wireless channel. First, station A starts the transmission process and it sends the packet after a DIFS and backoff period. After a short interframe space (SIFS), station A receives the ACK and the transmission process finishes. During station A's packet transmission, the timer of station B freezes. If station B wants to transmit at the end of station A's packet transmission, it has to wait for the channel to be idle for a DIFS. After the DIFS, the back-off timer is decremented until it reaches zero, and finally, station B sends the packet. This two-way handshaking technique for packet transmission is known as the basic access mechanism [8]. The standard defines as well a four-way handshaking mechanism ,which we do not consider in this work for simplicity of illustration.

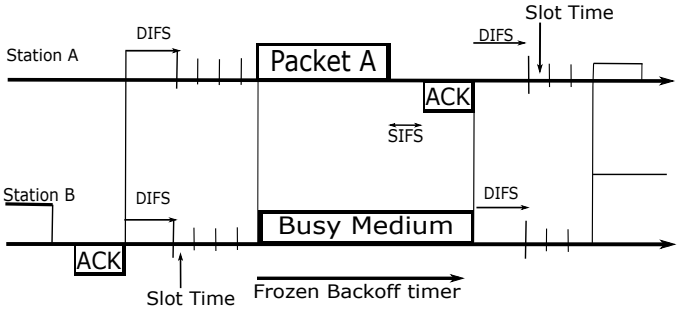


Fig. 1.1. 802.11 MAC protocol.

1.2 Throughput Optimization in WiFi Networks

The throughput in the 802.11 standard depends on the number of active stations and the contention window used by each station. If stations use a too small value of CW, then collisions will increase and as a consequence, the throughput performance will decrease. On the contrary, if stations use too large values of CW, the channel will be underutilized most of the time. Therefore the throughput performance will decrease. In general, keeping CW fixed, the throughput decreases as the number of stations in the wireless network increases. It has been shown in previous literature that there is an optimal value of CW that maximizes the throughput. The IEEE 802.11 standard does not specify the optimal value of the contention window but it is generally static and independent on the number of active stations in commercial wireless cards [9].

In multi-rate IEEE 802.11 WLANs, stations that use DCF and transmit at lower transmission rates make use of the channel for longer periods of time to transmit the same amount of data compared to stations using higher transmission rates. This reduces the throughput of high-rate stations in the WLAN, since less time

is available for transmission in the shared medium. This effect is known as performance anomaly. One solution to approach this problem is contention window tuning based on the proportional fair allocation of resources [10].

Consider a wireless network with two nodes, whose distances to the access point are not equal (node 1 is further and node 2 is closer to the access point). In case of transmission, node 1 will use more network resources than node 2 and this affects the performance of node 2 as well, for the reason that less channel time is available for node 2 transmissions. One solution in order to increase the efficiency and improve the performance of the network is to assume that the contention window of the nodes is such that the slot transmission probability (τ_1) of node 1 is assigned to 0 and the slot transmission probability of node 2 is assigned to 1. However, this approach cannot be considered a fair solution. Our aim is to formulate proportional fairness as a convex optimization problem whose objective is the sum of logs of throughputs maximization, which will intrinsically capture fairness while trying to achieve maximum performance. In our scenario, we assume that all the stations are saturated, (i.e. stations always have a packet to send). Let's $S_i(\tau)$ be the throughput of the station i, then:

$$S_i(\tau) = \frac{P_{\mathrm{succ},i} D_i}{\sigma P_{\mathrm{idle}} + T_c(1 - P_{\mathrm{idle}})}. \tag{1.1}$$

This formula is based on a renewal reward process.[2] Here two kinds of time slots are considered. The first one is the PHY idle slot duration without any transmission which is of duration σ.The second one is the busy slot which relates to the duration of a packet transmission [11]. The packet transmission interval is defined by T_c , which is the mean duration of a successful and, a collided transmission of node i or other stations' packet transmissions. T_c is considered equal for all kinds of former transmissions to simplify the analysis. The average packet size of the i_{th} station is defined by D_i in bits. $P_{\mathrm{succ},i}$ is the probability of a successful packet transmission of i_{th} station, which means that only station i transmits a packet while other stations are idle:

$$P_{\mathrm{succ},i} = \tau_i \prod_{k=1, k\neq i}^{n} (1 - \tau_k). \tag{1.2}$$

The term P_{idle} in the denominator of Eq.(1.1) is the probability that the channel is idle. When none of the stations attempt to transmit a packet, the probability is defined as the probability of no transmission [12]:

[2] A renewal process is an arrival process in which the interarrival intervals X_i are positive, independent and identically distributed (IID) random variables.

$$P_{\text{idle}} = \prod_{k=1}^{n}(1-\tau_k). \tag{1.3}$$

The term $1 - P_{\text{idle}}$ is the probability that the channel is busy for T_c duration due to the successful, unsuccessful (channel errors or collision) or other stations' packet transmissions [13]:

$$1 - P_{\text{idle}} = 1 - \prod_{k=1}^{n}(1-\tau_k). \tag{1.4}$$

Therefore, Eq.(1.1) is the amount of data transmitted per slot when that is successful over the average duration of a slot. It will be more useful to use the transformed variable $x_i = \frac{\tau_i}{(1-\tau_i)}$ rather than τ_i, $x_i \in [0, \infty)$ for $\tau_i \in [0, 1)$. Then the individual throughput changes to [12]:

$$S_i(x) = \frac{x_i}{X(x)}\frac{D_i}{T_c}. \tag{1.5}$$

Where $X(x) = a + \prod_{k=1}^{n}(1 + x_k) - 1$ with $a = \sigma/T_c$. As we consider n stations we can compute the aggregated throughput as below:

$$\bar{S}(x) = \sum_{i=1}^{n}\frac{x_i}{X(x)}\frac{D_i}{T_c}. \tag{1.6}$$

By considering proportional fairness our aim is to maximize the sum of the logs of throughputs [12]:

$$\begin{aligned} &\max. \quad \sum_{i=1}^{n}\tilde{S}_i(x), \\ &s.t. \quad \tilde{S}_i(x) \leq \log\frac{x_i D_i}{X(x)T_c}. \end{aligned} \tag{1.7}$$

The constraint certifies that the sum of logs of throughputs is feasible and sits in rate region Z. Since the log-transformed rate region $\tilde{Z}$ is strictly convex, there exists a unique solution that satisfies strong duality and (Karush-Kuhn-Tucker) KKT conditions which implies a global maximum [12].

CHAPTER 2. WiFi PROPORTIONAL FAIRNESS VIA BANDIT CONVEX OPTIMIZATION

In this section we briefly explain bandit convex optimization algorithms and present the proposed algorithm for the WiFi use case, called Online Gradient Descent with Sequential Multi-Point Gradient Estimates (OGD-SEMP).

2.1 Bandit Convex Optimization

In Bandit Convex Optimization (BCO) three steps are repeated between the player and the adversary. These three steps can be written for iteration t as follows:

- The player chooses a point $x_t \in K \subseteq R^d$.
- The adversary chooses a cost function $f_t \in F \subseteq R^k$.
- The player observes $f_t(x_t)$.

Here, x_t is a point from a fixed and known convex set. K represents a convex subset of a d-dimensional Euclidean space ($K \subseteq R^d$), however in this research d will be equal to 1. In addition, all the functions in F are convex [14]. The aim of this algorithm is to minimize the cumulative sum of the incurred losses. Thus, we can define regret after T iterations as follows:

$$R_T = \sum_{t=1}^{T} f_t(x_t) - \min_{x \in K} \sum_{t=1}^{T} f_t(x). \tag{2.1}$$

This formulation is known as cumulative regret, which measures the difference between the cumulative loss that is revealed to the player and the best-fixed decision in hindsight [1] [14].

Most of the BCO algorithms are based on Online Gradient Descent (OGD). The main goal and the most remarkable complication of BCO is to estimate the gradients of the cost functions. Therefore many researchers in BCO have investigated methods for estimating these gradients and using their results for designing algorithms for Bandit Convex Optimization (BCO) [1].

Zinkevich showed that a simple gradient descent strategy for the player incurs a $O(\sqrt{T})$ regret bound [15]. Flaxman et al. [16] proposed a scheme that combined the estimated gradients with the OGD algorithm of Zinkevich [17]. The algorithm of Flaxman et al. uses point evaluations of convex functions to approximately estimate the gradient. The regret bound of this algorithm is shown to be $O(T^{3/4})$. Agarwal et al. showed that in each round knowing the value of each cost

function at two points is almost as useful as knowing the value of each function everywhere, therefore their algorithm has a regret bound of $O(T^{2/3})$ improving the $O(T^{3/4})$ bounds achieved by Flaxman et al. However, Flaxman et al. and Agarwal et al. approaches cannot be used in a practical implementation and realistic setting in wireless networks. First, in many settings it is impossible to query cost functions two times in one iteration. Second, the variance of the single point estimators in the approach of Flaxman et al. [16] is large; consequently, speed of convergence is not practical for wireless stations [14][18].

2.2 Sequential Multi-Point Gradient Estimates in WiFi

We use a new multi-point BCO algorithm with a simpler assumption than that of Agarwal [15]. In this gradient estimation approach, queries are combined from two consecutive iterations. This algorithm is the modification of the multi-point gradient estimation and is called Online Gradient Descent with Sequential Multi-Point Gradient Estimates (OGD-SEMP) [14]. It includes a sequence of auxiliary points $y_1, y_2, ...$ which are used to keep track of the player's movement by updating gradient descent as follows:

$$y_{k+1} = \prod_{k}(y_k - \eta_k \tilde{g_k}). \tag{2.2}$$

Here, $\tilde{g_k}$ is the gradient estimate which is used to update y_{k+1}. The parameter η_k is the gradient descent step size and $\eta = \{\eta_1, \eta_2, \eta_3, ...\}$ is a sequence which shrinks over time. This coefficient defines the speed of convergence of gradient descent to the final value y_k. Fig. 2.1 shows a schematic of this algorithm for the k_{th} iteration. Let's consider y_k as the k_{th} point in a one-dimensional convex set with the interval $[y_k - \epsilon_k\delta_k, y_k + \epsilon_k\delta_k]$. Then, the distance between the beginning and the end of the interval is $2\epsilon_k\delta_k$. Here, ϵ_k is a random number that can be either -1 or 1. δ_k is a parameter which shrinks over time and it captures the distance between the selected point and y_k. In the first step, we choose an arbitrary point and obtain its cost function as:

$$\begin{aligned} \bar{x_t} &= y_k + \epsilon_k\delta_k, \\ {g_k}^+ &= f_t(\bar{x}_t). \end{aligned} \tag{2.3}$$

In the second step we choose another point in time $(t+1)$ and obtain its cost function as:

$$\begin{aligned} \bar{x}_{t+1} &= y_k - \epsilon_k \delta_k, \\ {g_k}^- &= f_{t+1}(\bar{x}_{t+1}). \end{aligned} \tag{2.4}$$

In the following step, the gradient can be estimated as follows:

$$\tilde{g_k} = \frac{{g_k}^+ - {g_k}^-}{2\epsilon_k \delta_k}. \tag{2.5}$$

Here, the numerator is the subtraction of two cost functions evaluations and the denominator is the distance between the beginning and the end of the interval (See Fig. 2.1), which corresponds to an unbiased estimator of the gradient [15].

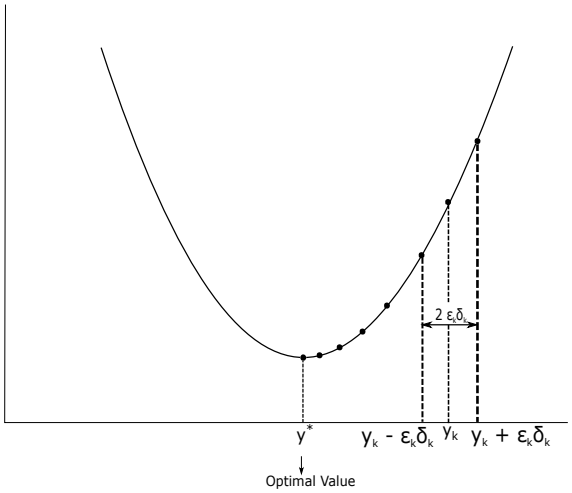


Fig. 2.1. Sketch of Online Gradient Descent with Sequential Multi-Point Gradient Estimates.

This algorithm was used in [14] for wireless networking optimization. They provided the theoretical analysis of this algorithm and evaluated its performance in an unlicensed LTE/WiFi fair coexistence use case.

Here we aim to use this method for achieving proportional fair allocation of resources in a wireless network and consequently maximize the sum of the logs of the throughput as seen in the previous section. Using this method, only by knowing the throughput of each station, regardless of other parameters of Eq.(1.7), we are able to achieve proportional fairness. Applied to the WiFi throughput proportional fairness case, the cost function is equal to the sum of the logs of the throughputs ($f_t = \sum_{i=1}^{n} \tilde{S}_i(x)$). Therefore, the cost functions in Eq.(2.3) and Eq.(2.4) ($f_t(\bar{x}_t)$, $f_{t+1}(\bar{x}_{t+1})$) define the sum of the logs of the throughputs at time t and time $t+1$ respectively, with $\bar{x}_t = \{x_1, x_2, ..., x_i\}$ a vector which defines x_i for all the stations at time t,[1] and $\bar{x}_{t+1} = \{x_1, x_2, ..., x_i\}$ a vector which defines point x

[1] Recall that x_i is a transformed variable $x_i = \frac{\tau_i}{(1-\tau_i)}$ rather than τ_i, $x_i \in [0, \infty)$ for $\tau_i \in [0, 1)$.

for all the stations at time $t+1$.

In more detail, consider a repeated game of T rounds. In each round $t = \{1, 2, ..., T\}$ we consider that WiFi network is formed by some number of WiFi stations using some channel access probabilities, modulation and coding rates (depending on varying channel conditions) and packet size. These parameters affect the WiFi throughput by impacting the transmission duration of a WiFi packet and the collision probability. Thus, in each round the WiFi network selects Eq.(1.5), which we now denote as f_t with some n, x_i, D_i and T_c. Then, in each round t:

• The WiFi access point chooses $\bar{x_t}$.

• The WiFi network (formed by all transmitting nodes) independently selects $f_t \in F$.

• The WiFi access point observes $f_t(\bar{x_t})$.

(as seen in section 1.2).

The precise algorithm is presented as Algorithm 1(extracted from [14]).

Algorithm 1 OGD SEMP

Input parameters: Non-increasing sequences $(\delta_k), (\eta_k)$.
Initialization: Choose arbitrary $y_0 \in K_{\delta_0}$.
For $k = 0, 1, \ldots,$ **repeat:**

1. Draw ε_k uniformly from $\{-1, 1\}$.
2. Let $t = 2k + 1$ and play
$$\bar{x}_t = y_k + \varepsilon_k \delta_k.$$
3. Set $g_k^+ = f_t(x_t)$.
4. Play
$$\bar{x}_{t+1} = y_k - \varepsilon_k \delta_k.$$
5. Set $g_k^- = f_{t+1}(x_{t+1})$.
6. Compute gradient estimate
$$\tilde{g}_k = \frac{g_k^+ - g_k^-}{2\varepsilon_k \delta_k}$$
7. Update
$$y_{k+1} = \Pi_k(y_k - \eta_k \tilde{g}_k),$$
where Π_k is the projection operator onto K_{δ_k}

CHAPTER 3. PERFORMANCE EVALUATION

3.1 Methodology

To evaluate the algorithm performance we resort to simulations. We use a custom simulator based on the IEEE 802.11 ac protocol. This custom simulator previously was used for evaluating OGD-SEMP performance in an unlicensed LTE/WiFi fair coexistence use case. In this research we extend the simulator to consider the particularities of the WiFi proportional fair use case gradient descent implementation. We will explain these changes in more detail at the end of this section.

The custom simulator consists of two main parts, the channel and the node module. The channel module connects to the node module through the physical layer and the mobility modules. It receives nodes' positions from the mobility module, receives data packets from the physical layer of a node and sends the packets to all nodes in the coverage range of the origins. In Fig. 3.1 these actions are named pos_in, from_phy and to_phy respectively.

The physical layer connects to the MAC layer as the upper layer, channel module as the lower layer and battery. This layer receives packets from the channel. This connection is defined as the from_channel. In the from_mac and the to_battery connections, this layer computes the energy of sending the packet, sets the transmitted power and sends it to the channel.

The next layer in Fig. 3.1 is the MAC layer which includes the OGD-SEMP algorithm and CW configurations. This layer connects to the network layer as its upper layer and the physical layer as its lower layer. In the from_phy connection, the MAC layer receives packets from the physical layer and detects the collisions. In the from_network, a new packet comes from a higher layer, is stored in the mac_queue if it is not full, then it computes the random backoff and sends the packet. In case of full queue the packet is dropped. All the MAC layer timers are set in this layer. The functionality of this layer strictly follows 802.11 operation and is explained in section 1.1 explicitly.

The network layer is responsible for the routing. According to the from_mac connection if the packet's destination matches with the defined node, it is sent to the application layer otherwise it is sent to the next hop. The parameters which are considered in the from_mac connection are source, destination, type and length of the received packet. In the from_transport connection a packet is received from the application layer and is sent to the next hop. In our case all nodes are directly connected, no routing is necessary.

The application layer is the last layer in the structure of the simulator. It generates the traffic of the simulations based on a Poisson distribution. Through the

from_network connection packets arrive from the network layer. The application layer also controls duplicated packets.

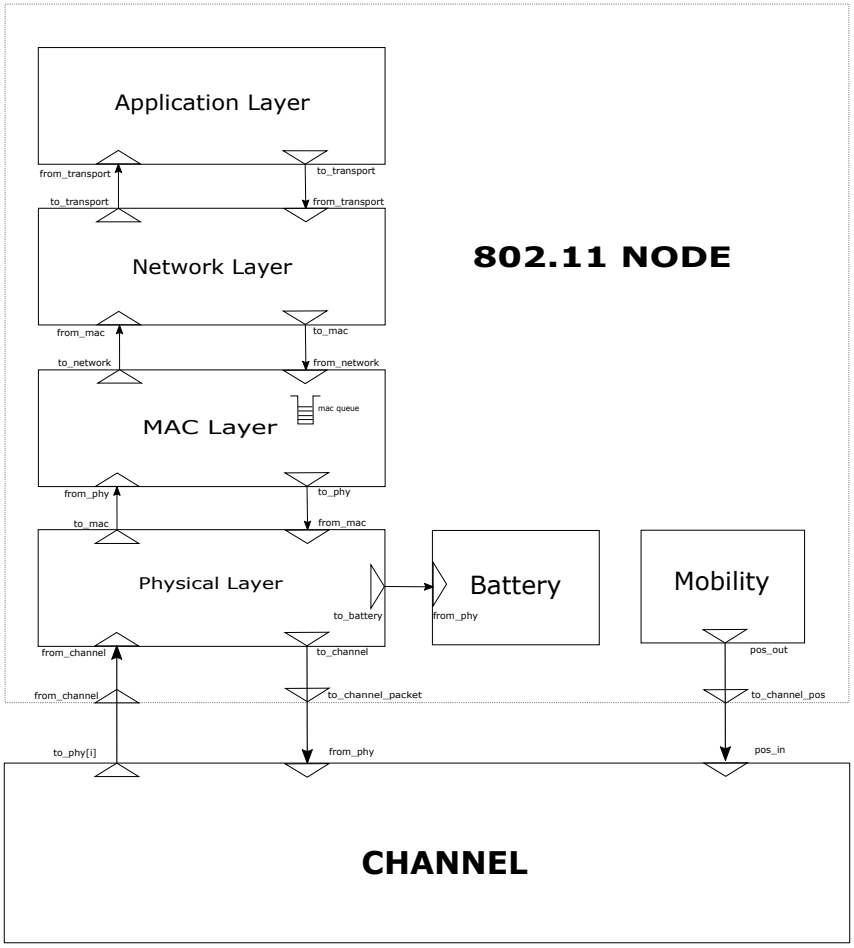


Fig. 3.1. Structure of the simulator.

All the parameters used by the simulator are listed in Table 3.1.

The packet transmission duration in the simulator is denoted by T_c. A successful transmission, includes the MAC ACK which is defined by T_{ack} , and T_{fra} which defines the duration of a data transmission. We express T_{ack} and T_{fra} by using the values of IEEE 802.11ac [19]:

$$\begin{aligned} T_{\text{fra}} &= T_{\text{plcp}} + \left\lceil \frac{L_s + n_{\text{agg}}(L_{\text{del}} + L_{\text{mac-h}} + D) + L_t}{n_{\text{sym}}} \right\rceil T_s, \\ T_{\text{ack}} &= T_{\text{plcp}} + \left\lceil \frac{L_s + L_{\text{ack}} + L_t}{n_{\text{sym}}} \right\rceil T_s. \end{aligned} \tag{3.1}$$

Here n_{sym} is the number of bits per OFDM symbol, T_s is the symbol duration and n_{agg} denotes the number of packets aggregated in a transmission. Finally, the value of T_c is defined as follows [20] (for simplicity we consider this duration equal

Table 3.1. Parameters of IEEE 802.11 ac [19].

Parameter	**Value**
Recv-Tx Delay	1 μs
Slot Duration (σ)	9 μs
DIFS	34 μs
SIFS	16 μs
PLCP Preamble + Header Duration (T_{plcp})	40 μs
EIFS	364 μs
TimerACK	314 μs
Propagation Time	1 μs
Tsymbol (T_s)	4 μs
PLCP Service Field (L_s)	16 bits
MAC Delimiter Field (L_{del})	32 bits
MAC Header ($L_{\mathrm{mac-h}}$)	288 bits
Tail Bits (L_t)	6 bits
ACK Length (L_{ack})	256 bits
Payload (D)	12000 bits
nsymbol (n_{sym})	1040 bits
number of aggregated packets (n_{agg})	64
Queue Size	1000
Retry Limit	1000
MIMO	4

for successful transmissions and collisions):

$$T_c = T_{\mathrm{fra}} + \mathrm{SIFS} + T_{ack} + \mathrm{DIFS}. \tag{3.2}$$

In order to implement the gradient descent algorithm for the WiFi proportional fair use case we have made some changes in the simulator. Since the contention window values are discrete while the slot transmission probabilities in our model are continuous, we need to convert discrete values of contention window to the desired continuous one. To achieve this we use a timer and two contention windows. During time interval t_1 the value of the contention window is set to CW_1 and during t_2 the value of the contention window is set to CW_2 (see Eq.(3.3)). The values of CW_1 and CW_2, t_1 and t_2 durations are configured in a way that their average is equal to the desired continuous one. Since the possible values for the contention window in commercial WiFi cards are $\{7, 15, 31, 63, 127, 255, 511, 1023\}$, we set CW_1 to the immediate lower value and CW_2 to the immediate higher value in the selected range. It is important to mention that when we change the value of CW as we are not resetting the backoff the random backoff which was ongoing will still use the previous value of CW. That will cause inaccuracies in the throughput estimation and generate noise, we will evaluate this in section 3.2.2. The relation between CW_1, CW_2, their durations and the continuous one is computed from

Eq.(3.3) and fixing $t_1 + t_2$.

$$\mathrm{CW} = t_1 \mathrm{CW}_1 + \mathrm{t}_2 \mathrm{CW}_2. \tag{3.3}$$

Note that the value of the continuous contention window (CW) is the reciprocal of the slot transmission probability (τ).

A second timer (that expires each T seconds) executes the gradient descent algorithm and computes the throughput estimation.

$$T = m(t_1 + t_2). \tag{3.4}$$

Eq.(3.4) shows that the gradient descent timer is a factor of the sum of the duration of t_1 and t_2 timers. In this equation the coefficient m is set to a positive integer. Fig. 3.2 shows a schematic of the contention window timer and gradient descent timer.

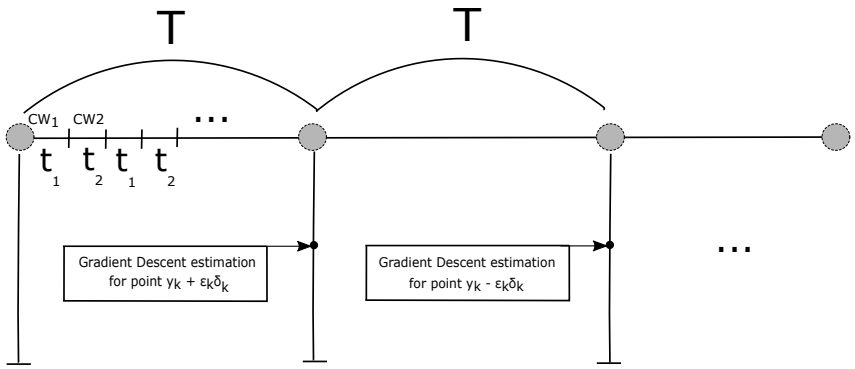


Fig. 3.2. Schematic of the different timers used to implement the gradient descent algorithm in the simulator.

3.2 Simulation Results

Here we present the evaluation of OGD-SEMP when applied to the WiFi use case by executing an extensive set of simulations in Matlab and in the simulator. We compare their performance using the individual throughput metric (S_t). We use Matlab in order to achieve the true values of the individual throughput computed using Eq.(1.7) for each set of experiments and use the simulator for evaluating the impact of having estimated values instead of the true value of the throughput. Noise in the estimates is caused by the random backoff, collision probabilities and discretization of the slot transmission probability as described above. By comparing the evolution of throughput over time for different settings we evaluate the algorithm's performance regarding the time to convergence.

In all simulations we have set the gradient descent step size as $\eta_k = \eta/k^{(1/2)}$ and $\delta_k = \omega/h(k)$, with ω as an input parameter and $h(k)$ as some increasing function. We will refer to ω as the exploration parameter and h as the exploration schedule. According to the number of stations in the network (n), the results are classified into four sets $(n = \{5, 10, 15, 20\})$. Note that stations always have a packet to transmit (nodes are saturated).

3.2.1 Evaluation of Convergence without Noise

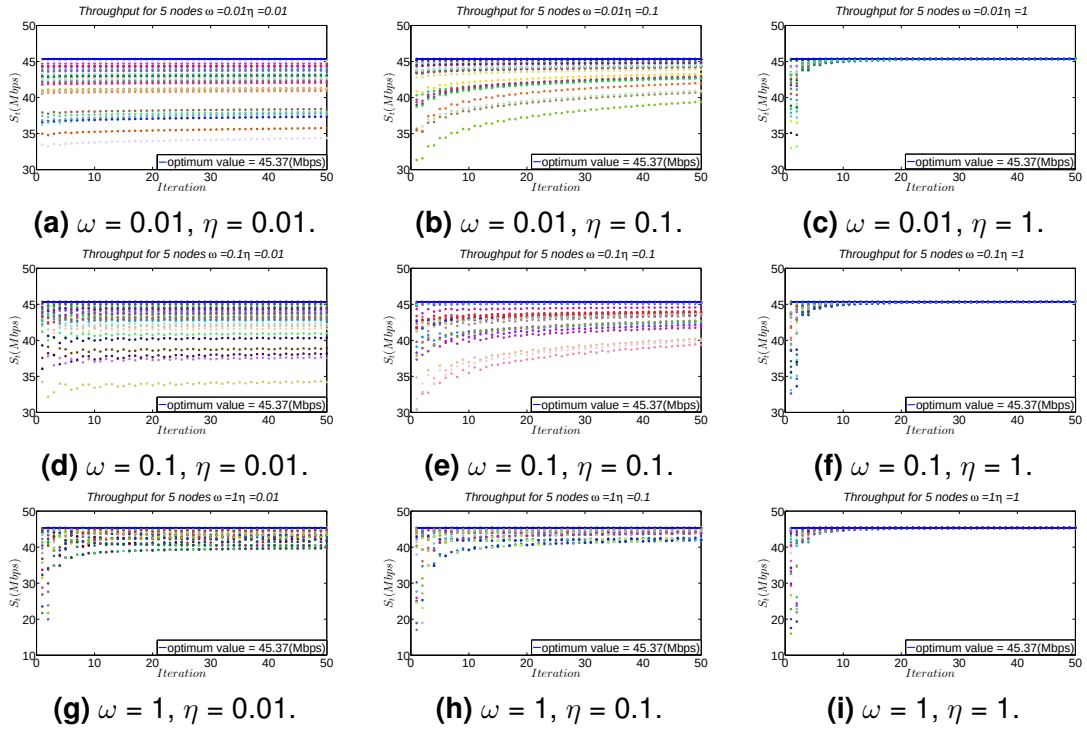
Our aim is to evaluate the sensitivity of the algorithm to the exploration parameter (ω), gradient descent step size (η_k) and different exploration schedules (h_k) by using the true value of the cost function. These results are not affected by estimation errors and we use continuous values for the slot transmission probability.

3.2.1.1 Sensitivity to the Learning Parameters

First, we evaluate the performance of the algorithm by changing the exploration parameter ω -the parameter that controls how far from y_k we take the two cost function evaluations at consecutive iterations- and gradient descent step size (η_k). Here $h(k)$ is considered equal to $k^{(3/4)}$ which shrinks the exploration parameter as time goes by. Fig. 3.3 shows the results of the individual throughput for 5 nodes during 50 iterations. We repeat the same simulations for 30 runs in order to obtain more accurate results. Optimal results [12] are shown in Fig. 3.3 as straight lines. The results are obtained from the implementation of the algorithm in Matlab with the cost function equal to the sum of the logs of the throughputs according to Eq.(1.7), IEEE 802.11ac parameters from Table 3.1. In order to keep the algorithm working between the minimum value of the contention window ($\mathrm{CW} = 7$) and its maximum value ($\mathrm{CW} = 1023$), we set the decision set as $[-6.930495, -1.945910]$ which are the minimum and the maximum values for the logs of the corresponding slot transmission probabilities. We vary the exploration parameter $\omega = \{0.01, 0.1, 1\}$ and gradient descent step size $\eta = \{0.01, 0.1, 1\}$.

Fig. 3.3 shows that by fixing η and increasing parameter ω the rate of convergence increases. As we saw in Eq.(2.3) increasing the value of exploration parameter (ω), we take bigger steps towards the optimum. By fixing parameter ω and increasing the parameter η for the range of values considered the rate of convergence increases as well since we also make larger steps with bigger gradient descent step sizes. We observe that the increasing trend in the second case is faster than the first case. It can be seen that for exploration parameter equal to $\{0.01, 0.1, 1\}$ and gradient descent step size equal to 1 the algorithm converges to the optimum value after a few numbers of iterations (less than 20 iterations).

Here we keep the algorithm setup same as above and increase the number of



(a) ω = 0.01, η = 0.01. **(b)** ω = 0.01, η = 0.1. **(c)** ω = 0.01, η = 1.

(d) ω = 0.1, η = 0.01. **(e)** ω = 0.1, η = 0.1. **(f)** ω = 0.1, η = 1.

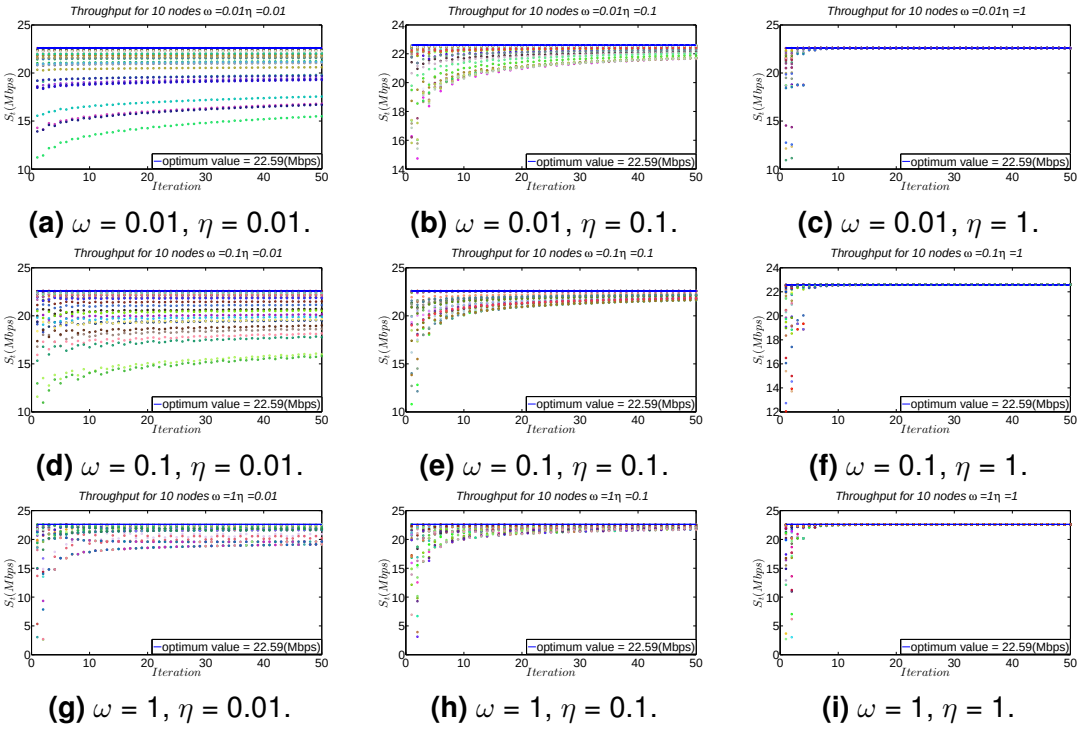
(g) ω = 1, η = 0.01. **(h)** ω = 1, η = 0.1. **(i)** ω = 1, η = 1.

Fig. 3.3. Individual throughput for $n = 5$ using different ω, η and $h(k) = k^{(3/4)}$.

nodes. Fig. 3.4 to 3.6 illustrate the individual throughput of a network with 10, 15 and 20 nodes respectively. We observe that for the same value of ω and η increasing the number of nodes, the algorithm converges to the optimum faster. The reason for this behavior is illustrated in Fig. 3.7. This figure shows the shape of the convex function [12] for the different number of nodes. As it is shown by increasing the number of the nodes in the network the function becomes steeper, thus the gradients are larger. This means that the algorithm makes larger steps at each iteration and reaches to the optimum value faster. Note that the difference between the minimum value of the convex function and its maximum is increased by increasing the number of nodes. For this case the algorithm converges in around 10 iterations or less for $\omega = \{0.01, 0.1, 1\}$ and $\eta = 1$.

3.2.1.2 Sensitivity to the Exploration Schedules

In this set of simulations we use the same setup as in section 3.2.1.1 and change the exploration schedule to $h(k) = k^{(1/2)}$. Here we evaluate the sensitivity of the algorithm to two different exploration schedules. Similarly to Fig. 3.3, Fig. 3.8 shows for different values of ω, η and $h(k) = k^{(1/2)}$ the individual throughput. We can observe that the convergence speed is almost the same as in Fig. 3.3. Only for the case $n = 5$, $\omega = 1$ and $\eta = 1$ (Fig. 3.3(i) and Fig. 3.8(i)) we observe that the convergence speed in Fig. 3.3(i) with $h(k) = k^{(3/4)}$ is faster than the one with $h(k) = k^{(1/2)}$. These results show that the sensitivity of the algorithm to the exploration schedule is negligible for the exploration schedules considered.

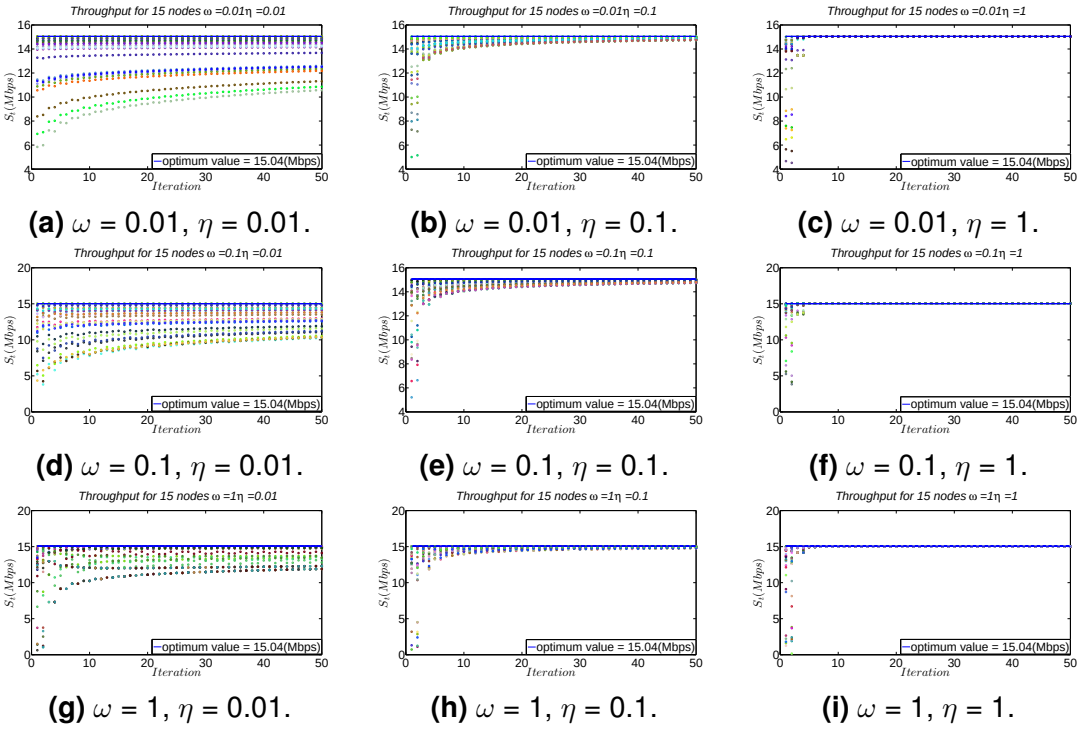


(a) ω = 0.01, η = 0.01. **(b)** ω = 0.01, η = 0.1. **(c)** ω = 0.01, η = 1.

(d) ω = 0.1, η = 0.01. **(e)** ω = 0.1, η = 0.1. **(f)** ω = 0.1, η = 1.

(g) ω = 1, η = 0.01. **(h)** ω = 1, η = 0.1. **(i)** ω = 1, η = 1.

Fig. 3.4. Individual throughput for $n = 10$ using different ω, η and $h(k) = k^{(3/4)}$.

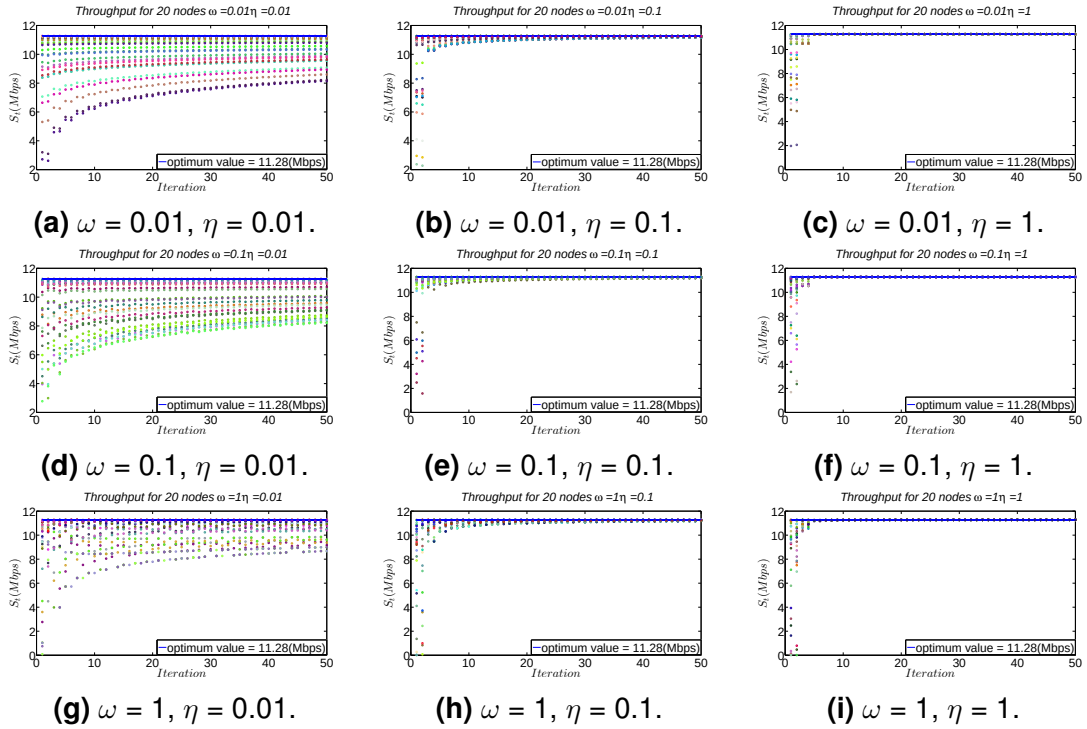


(a) ω = 0.01, η = 0.01. **(b)** ω = 0.01, η = 0.1. **(c)** ω = 0.01, η = 1.

(d) ω = 0.1, η = 0.01. **(e)** ω = 0.1, η = 0.1. **(f)** ω = 0.1, η = 1.

(g) ω = 1, η = 0.01. **(h)** ω = 1, η = 0.1. **(i)** ω = 1, η = 1.

Fig. 3.5. Individual throughput for $n = 15$ using different ω, η and $h(k) = k^{(3/4)}$.

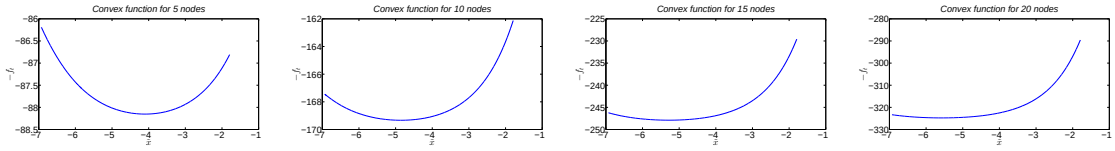


(a) ω = 0.01, η = 0.01. **(b)** ω = 0.01, η = 0.1. **(c)** ω = 0.01, η = 1.

(d) ω = 0.1, η = 0.01. **(e)** ω = 0.1, η = 0.1. **(f)** ω = 0.1, η = 1.

(g) ω = 1, η = 0.01. **(h)** ω = 1, η = 0.1. **(i)** ω = 1, η = 1.

Fig. 3.6. Individual throughput for $n = 20$ using different ω, η and $h(k) = k^{(3/4)}$.



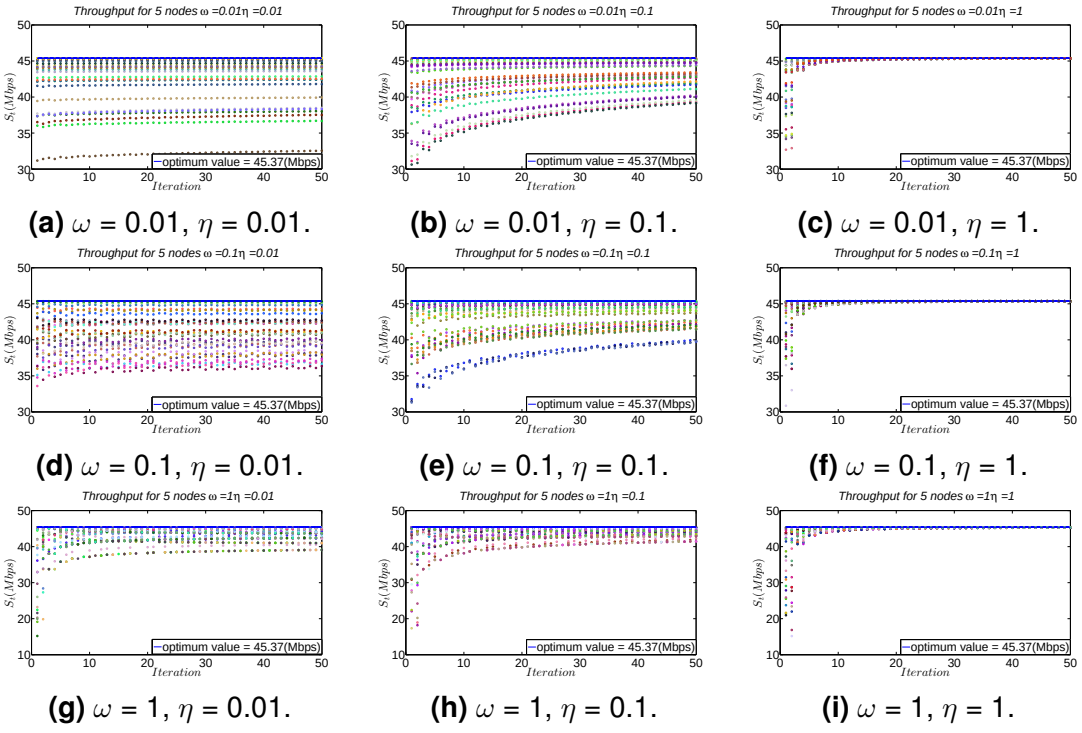
(a) $n = 5$. **(b)** $n = 10$. **(c)** $n = 15$. **(d)** $n = 20$.

Fig. 3.7. Shape of the convex functions for different number of nodes.

This means that the impact of $h(k)$ compared to the gradient descent step size η, exploration parameter ω and gradients is low. We also see, as before, that by increasing the number of nodes in the network the individual throughput value converges to its optimum value faster (See Fig. 3.9, 3.10 and 3.11).

3.2.2 Evaluation of Convergence Adding Noise

In this section first, we assess noisy estimates of throughput from the simulator. Then we evaluate the impact of the discretization of the slot transmission probability timers $(t_1 + t_2)$ and gradient descent timer (T) on this noise.

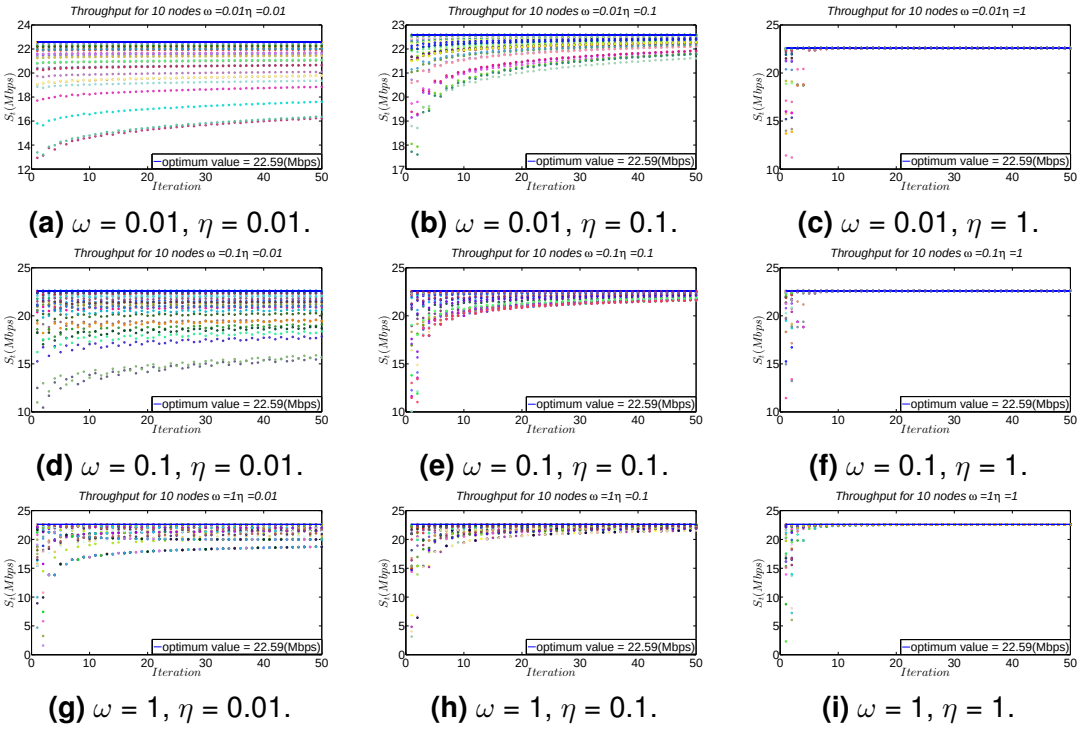


(a) ω = 0.01, η = 0.01. **(b)** ω = 0.01, η = 0.1. **(c)** ω = 0.01, η = 1.

(d) ω = 0.1, η = 0.01. **(e)** ω = 0.1, η = 0.1. **(f)** ω = 0.1, η = 1.

(g) ω = 1, η = 0.01. **(h)** ω = 1, η = 0.1. **(i)** ω = 1, η = 1.

Fig. 3.8. Individual throughput for $n = 5$ using different ω, η and $h(k) = k^{(1/2)}$.

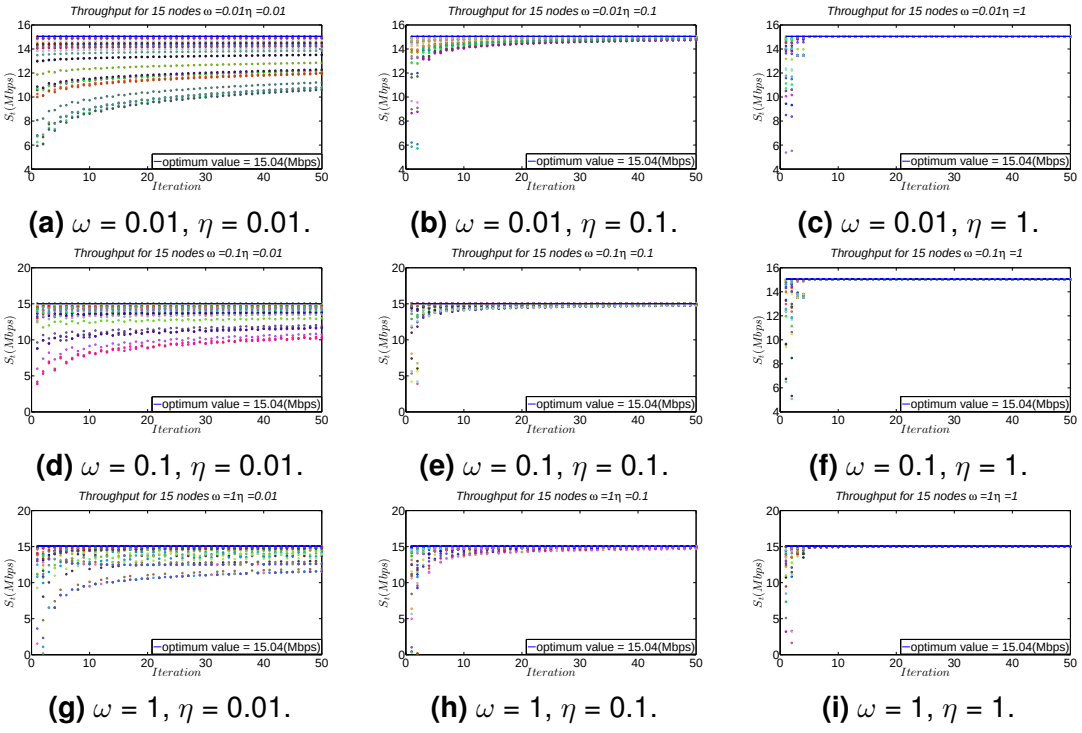


(a) ω = 0.01, η = 0.01. **(b)** ω = 0.01, η = 0.1. **(c)** ω = 0.01, η = 1.

(d) ω = 0.1, η = 0.01. **(e)** ω = 0.1, η = 0.1. **(f)** ω = 0.1, η = 1.

(g) ω = 1, η = 0.01. **(h)** ω = 1, η = 0.1. **(i)** ω = 1, η = 1.

Fig. 3.9. Individual throughput for $n = 10$ using different ω, η and $h(k) = k^{(1/2)}$.

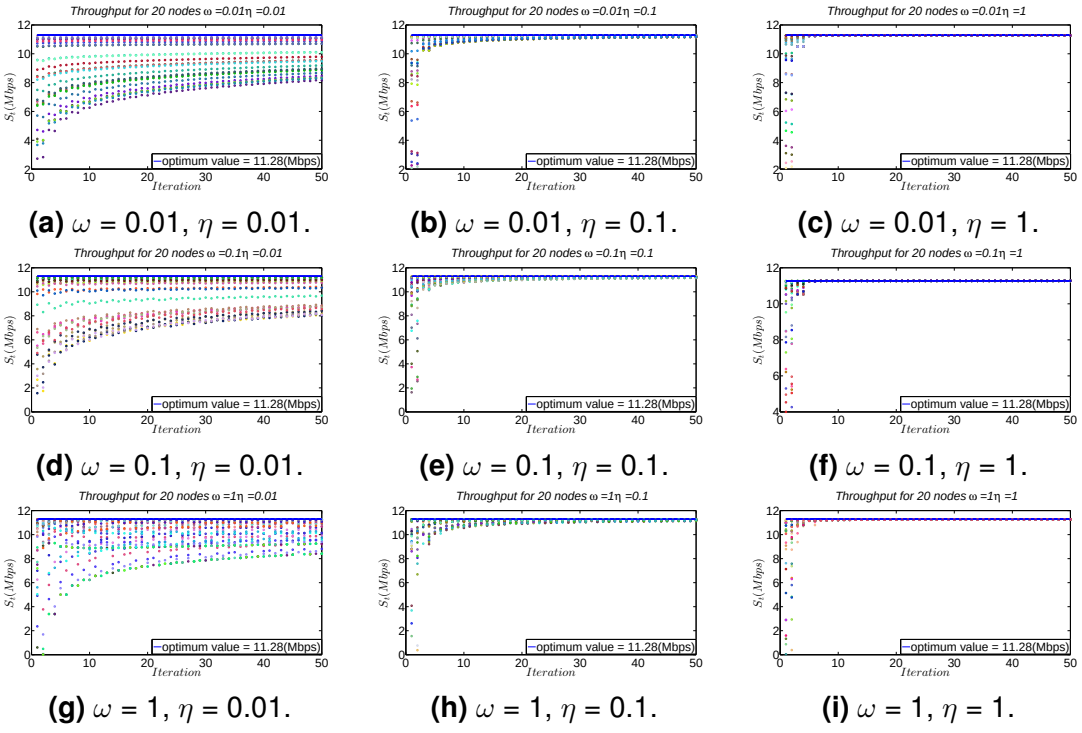


(a) ω = 0.01, η = 0.01. **(b)** ω = 0.01, η = 0.1. **(c)** ω = 0.01, η = 1.

(d) ω = 0.1, η = 0.01. **(e)** ω = 0.1, η = 0.1. **(f)** ω = 0.1, η = 1.

(g) ω = 1, η = 0.01. **(h)** ω = 1, η = 0.1. **(i)** ω = 1, η = 1.

Fig. 3.10. Individual throughput for $n = 15$ using different ω, η and $h(k) = k^{(1/2)}$.

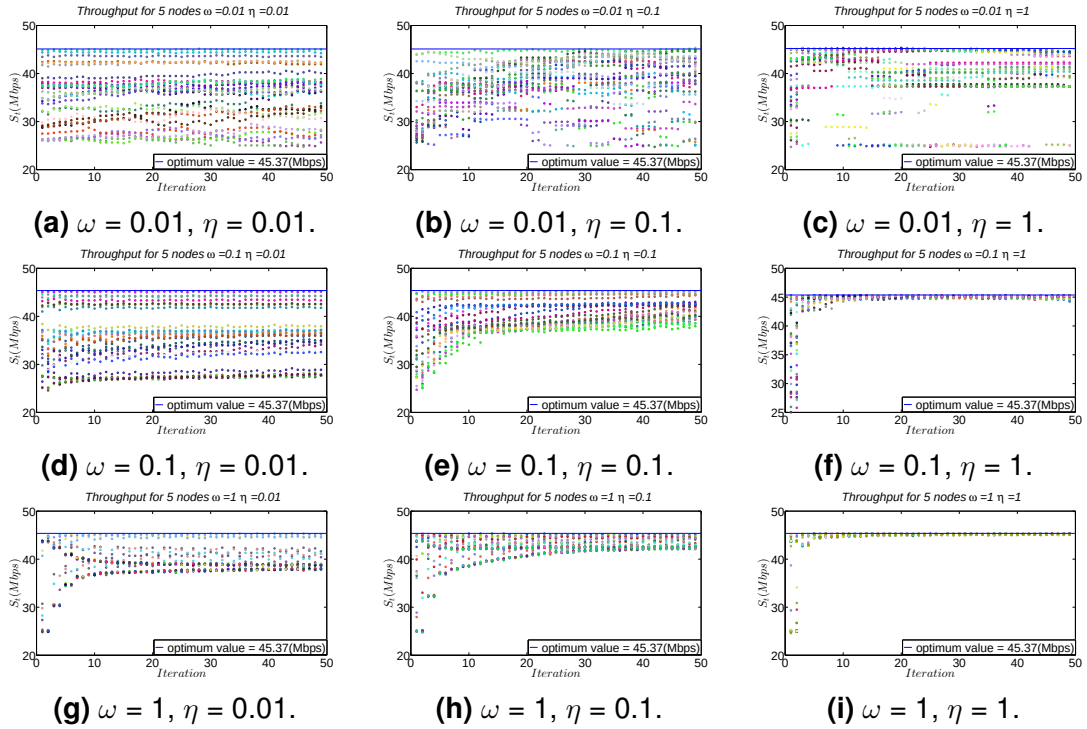


(a) ω = 0.01, η = 0.01. **(b)** ω = 0.01, η = 0.1. **(c)** ω = 0.01, η = 1.

(d) ω = 0.1, η = 0.01. **(e)** ω = 0.1, η = 0.1. **(f)** ω = 0.1, η = 1.

(g) ω = 1, η = 0.01. **(h)** ω = 1, η = 0.1. **(i)** ω = 1, η = 1.

Fig. 3.11. Individual throughput for $n = 20$ using different ω, η and $h(k) = k^{(1/2)}$.



(a) ω = 0.01, η = 0.01. **(b)** ω = 0.01, η = 0.1. **(c)** ω = 0.01, η = 1.

(d) ω = 0.1, η = 0.01. **(e)** ω = 0.1, η = 0.1. **(f)** ω = 0.1, η = 1.

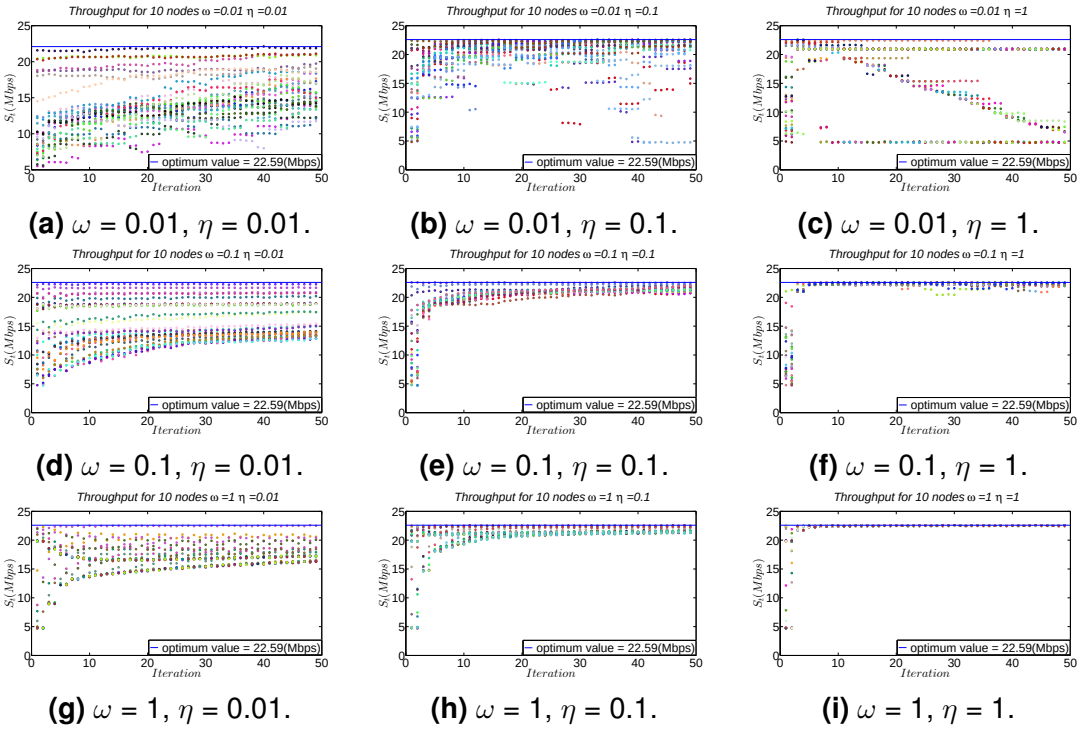
(g) ω = 1, η = 0.01. **(h)** ω = 1, η = 0.1. **(i)** ω = 1, η = 1.

Fig. 3.12. Individual throughput for $n = 5$ using different ω, η and $h(k) = k^{(3/4)}$ (Simulator).

3.2.2.1 Sensitivity to Noisy Gradient Estimates

Here we evaluate the performance of the algorithm by having noisy estimates of the individual throughput instead of the true value. In order to achieve this goal we implement the algorithm in the simulator which is explained in section 3.1 previously. We set the exploration parameter to $\omega = \{0.01, 01, 1\}$ and gradient descent step size to $\eta = \{0.01, 01, 1\}$. We also set the gradient descent timer equal to $T = 100$ s and the value of contention window timer equal to $(t_1+t_2) = 0.1$ s. Each simulation is repeated for 30 runs in order to achieve more accurate results. Here the exploration schedule is set to $k^{(3/4)}$.

Fig. 3.12 to 3.15 show the results of the individual throughput for 5, 10, 15, 20 nodes respectively during 50 iterations. These figures illustrate that by increasing the exploration parameter (ω) the convergence becomes faster and the effect of the noise in the individual throughput estimation is reduced. Note that the algorithm still converges in less than 10 iterations for $\omega = 1$ and $\eta = 1$. We see that for $\omega = 0.01$ the individual throughputs are not following the desired convergence trend. The reason for this behavior is the noise: with a small value of ω the gradient estimations are less accurate making more probable for gradient descent to move in the opposite direction. We evaluate this in more detail in the following section.

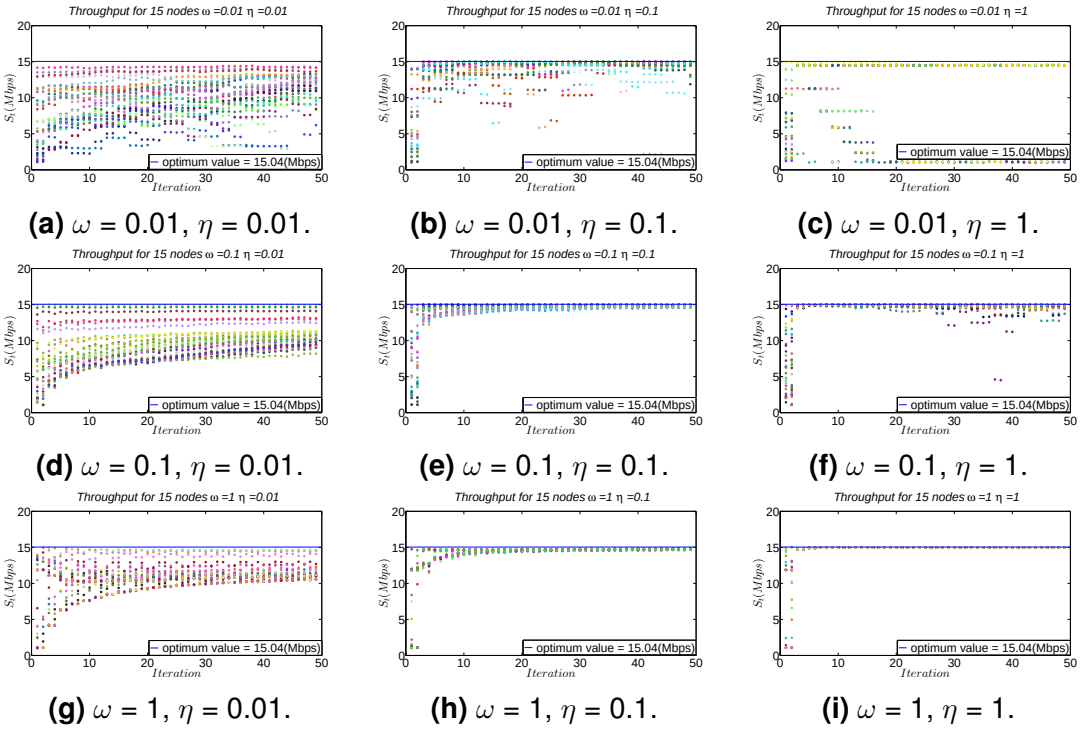


(a) ω = 0.01, η = 0.01. **(b)** ω = 0.01, η = 0.1. **(c)** ω = 0.01, η = 1.

(d) ω = 0.1, η = 0.01. **(e)** ω = 0.1, η = 0.1. **(f)** ω = 0.1, η = 1.

(g) ω = 1, η = 0.01. **(h)** ω = 1, η = 0.1. **(i)** ω = 1, η = 1.

Fig. 3.13. Individual throughput for $n = 10$ using different ω, η and $h(k) = k^{(3/4)}$ (Simulator).

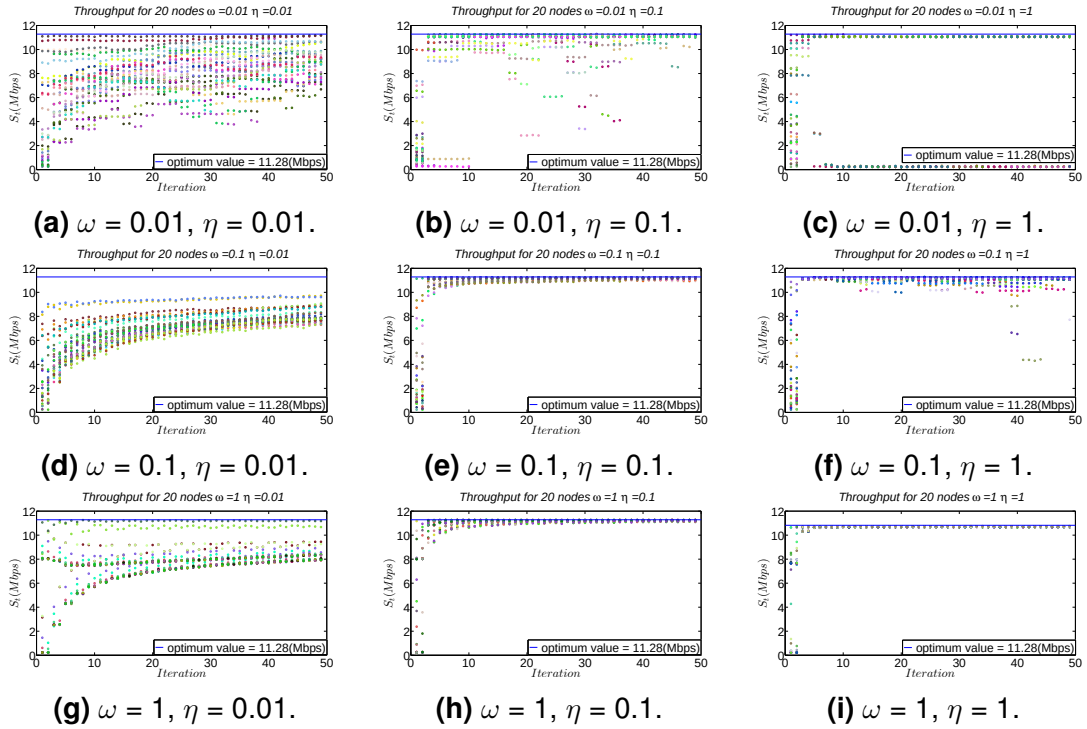


(a) ω = 0.01, η = 0.01. **(b)** ω = 0.01, η = 0.1. **(c)** ω = 0.01, η = 1.

(d) ω = 0.1, η = 0.01. **(e)** ω = 0.1, η = 0.1. **(f)** ω = 0.1, η = 1.

(g) ω = 1, η = 0.01. **(h)** ω = 1, η = 0.1. **(i)** ω = 1, η = 1.

Fig. 3.14. Individual throughput for $n = 15$ using different ω, η and $h(k) = k^{(3/4)}$ (Simulator).



(a) ω = 0.01, η = 0.01. **(b)** ω = 0.01, η = 0.1. **(c)** ω = 0.01, η = 1.

(d) ω = 0.1, η = 0.01. **(e)** ω = 0.1, η = 0.1. **(f)** ω = 0.1, η = 1.

(g) ω = 1, η = 0.01. **(h)** ω = 1, η = 0.1. **(i)** ω = 1, η = 1.

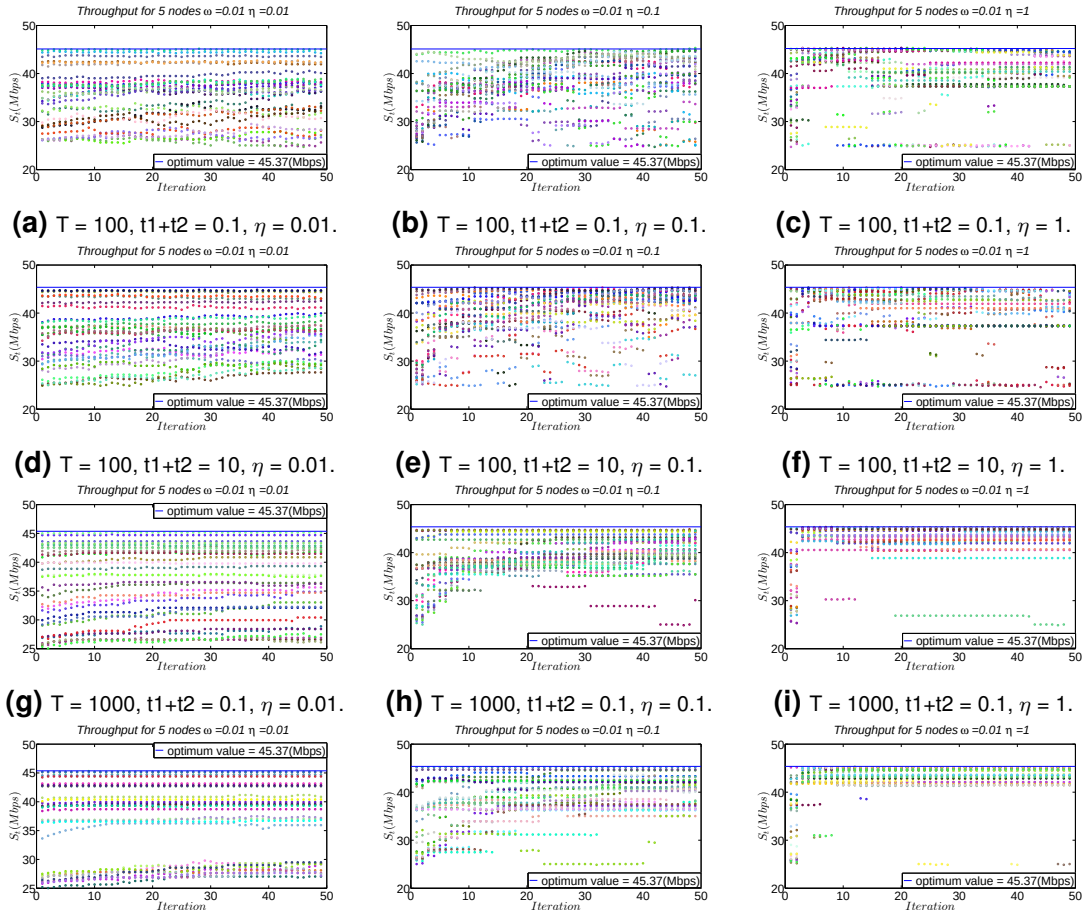
Fig. 3.15. Individual throughput for $n = 20$ using different ω, η and $h(k) = k^{(3/4)}$ (Simulator).

3.2.2.2 Sensitivity to Different Temporal Batches

In order to see which parameters affect the accuracy of the simulations and control the effect of noise we change the duration of the contention window timer and gradient descent timer while setting the exploration parameter to 0.01, gradient descent step size to $\{0.01, 0.1, 1\}$ and the exploration schedule to $k^{(3/4)}$.

By increasing the duration of the gradient descent timer from $T = 100$ s to $T = 1000$ s the algorithm convergence improves slightly (Fig. 3.16a-c and Fig. 3.16g-i). The reason for this is explained by the noisy convex function with different values of gradient descent timer, $T = 100$ s and $T = 1000$ s respectively. In the case of $T = 1000$ s convex functions' estimations are closer to the true value (see Fig. 3.7). Hence, there is a trade-off between the accuracy of the results and time to convergence (the algorithm takes longer to converge when we wait 1000 s at each iteration).

We argue that by selecting small ω, the results are affected more by the noise thus, convergence speed is worse than the ones with larger values of the exploration parameter. By comparing Fig. 3.16g-i and Fig. 3.16j-l we observe that the effect of changing the contention window timer from $t_1 + t_2 = 0.1$ s to $t_1 + t_2 = 10$ s is negligible. This means that the effect of having ongoing backoff countdown using the previous value of CW does not affect noise considerably.



(a) T = 100, t1+t2 = 0.1, η = 0.01. **(b)** T = 100, t1+t2 = 0.1, η = 0.1. **(c)** T = 100, t1+t2 = 0.1, η = 1.

(d) T = 100, t1+t2 = 10, η = 0.01. **(e)** T = 100, t1+t2 = 10, η = 0.1. **(f)** T = 100, t1+t2 = 10, η = 1.

(g) T = 1000, t1+t2 = 0.1, η = 0.01. **(h)** T = 1000, t1+t2 = 0.1, η = 0.1. **(i)** T = 1000, t1+t2 = 0.1, η = 1.

(j) T = 1000, t1+t2 = 10, η = 0.01. **(k)** T = 1000, t1+t2 = 10, η = 0.1. **(l)** T = 1000, t1+t2 = 10, η = 1.

Fig. 3.16. Individual throughput for $n = 5$ by changing gradient descent timer and contention window timers.

To sum up, we have seen that by selecting proper values for the algorithm's learning parameters (exploration parameter and gradient descent step size), it converges to the optimum value in a few number of iterations. We have also observed that the proposed algorithm's performance is remarkably affected by the exploration parameter rather than the exploration schedule, and that parameter has a more severe impact on the algorithm performance when we do not have access to the true value of the function but to noisy estimates. We also showed that the impact of noisy function estimates on the results can be alleviated by increasing the duration of the temporal batch during which we perform the function estimation.

CHAPTER 4. RELATED WORK

The first approaches to improve throughput in wireless local area networks were based on MAC parameter tuning. Although there are many tuning algorithms in the literature, here we refer to those which are presented by Serrano et al. [21], Siris et al. [22], Freitag et al. [23], and Bellalta et al. [24] as representative examples.

For instance, adaptive $\mathrm{CW}_{\min}$ which is presented by Serrano et al. [21] is one of the simplest algorithms. Since all the MAC parameters are fixed except $\mathrm{CW}_{\min}$ of nonsensitive flows, this algorithm is called quasi static configuration of the (Enhanced Distributed Channel Access) EDCA parameters. Their proposed algorithm gives a priority to VoIP flows rather than elastic flows, thus it is very restrictive with elastic flows. In this algorithm $\mathrm{CW}_{\min}$ is a variable which depends on the number of active stations in the network (n). Also, it provides a degree of accuracy of adaptation according to the AP state.

Another paradigm is defined by Siris et al. [22]. In this algorithm contention window increments iteratively to obtain the optimal configuration of the $\mathrm{CW}_{\min}$ parameter and consecutively maximize the access point throughput. Increasing the $\mathrm{CW}_{\min}$ parameter reduces the channel collision hence the throughput increases in some cases.

A more comprehensive algorithm is presented by Freitag et al. [23]. In this approach the $\mathrm{CW}_{\min}$ and TXOP (which controls the amount of packets that can be transmitted back-to-back at each transmission attempt) parameters are tunned based on the throughput and the number of the stations (n). Since it sets the TXOP value proportionally to the bandwidth, it provides a fairer channel occupation. Also, by tuning the $\mathrm{CW}_{\min}$ value, it is able to control the collision probability.

The last one that we have mentioned above is proposed by Bellalta et al. [24]. In this algorithm parameters such as CWmin, TXOP and AIFSN (setting DIFS differently at different stations to control which ones access the channel first) are tuned to reach the best configuration for maximizing the throughput and the bandwidth requirements at the same time. It is based on the set of optimal values of the MAC parameters from a maximization process. This algorithm computes iteratively the best parameters given the WLAN status.

The common MAC parameter which is tunned in all these algorithms is contention window, as it effect is more pronounced. Static algorithms define all the MAC parameters and keep them fixed. They cannot adapt themselves to the network changes hence they are inefficient. Whereas adaptive algorithms such as the ones we have overviewed are able to set the MAC parameters accordingly to the network status. The algorithms that consider all the parameters [23, 24] achieve a more accurate tuning yet they are still suboptimal.

On one hand, the solutions from iterative algorithms [22, 24] are more accurate

regarding the optimal solutions, but they require long time to compute the final solution. Therefore they are not practical in real-time use cases. On the other hand, non-iterative algorithms [21, 23] work in real-time but are not as accurate as iterative ones.

The most recent works on WiFi network throughput optimization are based on a proportional fairness approach. There are many works on this approach in the literature. The reader is referred to those which are presented by Checco et al. [12], Patras et al. [25] and Valls et al. [26], which are similar in nature.

Checco et al. [12] pioneered rigorous analysis of proportional fairness in IEEE 802.11 WLANs. They proved that a unique proportional fair rate allocation exists as the flow total air-time. This algorithm corrects previous works on air-time quantities and uses the IEEE 802.11 rate region as a log-convex. It satisfies per station fairness and per flow fairness. In these approaches [12, 25, 26], throughput optimization is achieved by inferring MAC parameters and network metrics such as packet transmission duration, slot transmission probability and average packet size of the stations. Therefore these approaches are not easy to implement. These metrics can be estimated but we need to handle estimation errors and network dynamics.

We base our algorithm on these rigorous approaches but without the need to know all parameters of the function to optimize (thus without the need to infer MAC parameters and network metrics such as packet transmission duration, slot transmission probability and average packet size of the stations). Our interest is in evaluating time of convergence and adaptability to changes. Commercial WiFi cards do not implement any of these algorithms. One reason for this may be the need to estimate network parameters. Therefore the presented algorithm aims to be practical, more accurate, simple to implement and applicable in real-time.

CHAPTER 5. CONCLUSION

In this chapter we summarize the main contributions of this thesis and, describe some directions for future work, sustainability and ethical considerations.

5.1 Contributions

The main contribution of this research is on achieving proportional fairness in WiFi networks by applying convex bandit optimization. We have applied the OGD-SEMP algorithm based on the BCO algorithm to the WiFi proportional fairness use case. BCO is a powerful framework in wireless network optimization problems, because it is capable of throughput optimization in wireless networks without the need to estimate network parameters and adapts to changes in the network intrinsically. These two characteristics are appealing to network optimization application in real deployments.

According to the results we showed, with the appropriate setting of parameters, the algorithm converges to the optimum value in a few number of iterations. However, the parameter of the algorithm that controls its exploration has a significant impact on the algorithm's performance, especially when we are faced with throughput estimation errors. This can be alleviated by increasing the duration of the estimation periods but at the cost of longer convergence times. Our results show that the algorithm is a practical solution for wireless network optimization, but that care has to be taken when configuring the algorithm parameters.

In the previous work, we set up a real wireless network that included one access point and five stations. We tried to measure the individual throughput in the access point while varying the value of the contention window. The goal was to perform online adaptation of the contention window to enable implementation of the algorithm in real hardware with on demand contention window dissemination by the access point as described in [25], see section 5. However, after troubleshooting we realized that the hardware we were using as stations (Raspberry Pi's 3) neglected the beacon configurations provided in beacons. Real hardware implementation of the algorithm was left to future work as described in the section below.

5.2 Future Work

It could be interesting to extend this investigation with some other simulations and experiments such as follows:

Introduce a method to reduce the effect of noise in the results of the custom simulator in a more efficient way. For instance, a proposed solution will be using averages of the gradient estimations. In this method the estimations of the gradients will be closer to the true value while the algorithm will not take longer to converge. Therefore in general the algorithm will be more efficient.

One possible future work could be evaluating the algorithm's adaptability to the network dynamics in the custom simulator. This can be achieved by changing the number of the WiFi stations (n) during gradient descent computation. We expect that the proposed algorithm will rapidly adapt to slow changes in the number of WiFi station, as seen in [14] for the unlicensed LTE/WiFi coexistence use case. In [14] the authors showed that the algorithm reacts slower to recent changes. The reason for this is the reduction of both gradient descent step size (η_k) and δ_k. Conversely, in case of the sudden changes and faster dynamics the algorithm is still able to converge to the optimum value. However, algorithm's adaptation to the new setting takes longer.

Another future research line is demonstrating the proposed algorithm in a real wireless network scenario. One option to achieve this will be using hardware such as PC engine APU as the access point and the stations in the network. In this scenario the proposed algorithm will be implemented only in the access point. Then all the stations will connect to the AP as clients. In order to measure the individual received throughput in the AP, the Iperf tool can be used in the AP. The configuration of the CW and individual throughput measurements for all the stations will be done as well in the AP. Therefore in order to reach the optimum value of the individual throughput according to the number of the stations in the network, the AP can automatically vary the values of CW and communicate it to the stations via beacons.

Also, it could be interesting to evaluate the performance of the proposed algorithm during certain cyber-attacks that have significant impact on the network performance and create moderate volumes of traffic. Since this algorithm optimizes the throughput, in case of attack -for those which are less harmful than jamming- we expect an improvement in the throughput. Note that in case of a jamming attack the frequency drops the signal to a level where the wireless network can not longer function, thus it is not feasible to monitor the effect of this attack on the performance of the algorithm. However, for attacks whose aim is to reduce network capacity and reduce the Quality of Service, the proposed algorithm can help in alleviating their effects by being more efficient with the available network resources.

5.3 Sustainability Consideration

Since the aim of this research is on introducing a solution for wireless network's performance improvement, we employ a BCO algorithm to achieve proportional fair resource allocation in wireless networks. Since the unlicensed spectrum is a scarce resource by using the proposed algorithm, we believe that stations will use the spectrum in a fair, more efficient way.

5.4 Ethical Consideration

The proposed algorithm is introduced as a practical and easy-to-implement solution for commercial WiFi cards. Since by implementing this algorithm WiFi networks become more efficient it is possible to integrate more applications in a network. Consequently, and in general we argue that mechanisms on throughput optimization can have a positive impact on economy.

Since in this research we only worked on the throughput optimization in WLANs and we did not contrive any personal data, through wireless network it can be declared that this project does not violate data protection law.

ACRONYMS

ACK	Acknowledgment
AIFSN	Arbitration Inter-Frame Space Number
AP	Access Point
APU	Auxiliary Power Unit
BCO	Bandit Convex Optimization
CSMA/CA	Carrier Sense Multiple Access/Collision Avoidance
CW	Contention Window
DCF	Distributed Coordination Function
DIFS	Distributed Interframe Space
EDCA	Enhanced Distributed Channel Access
KKT	Karush-Kuhn-Tucker
LTE	Long Term Evaluation
MAC	Media Access Control
MMSE	Multimedia Messaging Service Environment
OCO	Online Convex Optimization
OFDM	Orthogonal Frequency-Division Multiplexing
OGD	Online Gradient Descent
OGD-SEMP	Online Gradient Descent with Sequential Multi-Point Gradient Estimates
SIFS	Short Interframe Space
TXOP	Transmission Opportunity
WiFi	Wireless Fidelity
WLAN	Wireless Local Area Network

Bibliography

[1] Elad Hazan et al. Introduction to Online Convex Optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.

[2] Timothy N Davidson. Enriching the Art of FIR Filter Design via Convex Optimization. *IEEE signal processing magazine*, 27(3):89–101, 2010.

[3] D Perez Palomar, John M Cioffi, and Miguel Angel Lagunas. Joint Tx-Rx Beamforming Design for Multicarrier MIMO Channels: A Unified Framework for Convex Optimization. *IEEE Transactions on Signal Processing*, 51(9):2381–2401, 2003.

[4] David Julian, Mung Chiang, Daniel O'Neill, and Stephen Boyd. QoS and Fairness Constrained Convex Optimization of Resource Allocation for Wireless Cellular and Ad Hoc Networks. In *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 477–486. IEEE, 2002.

[5] Ami Wiesel, Yonina C Eldar, and Shlomo Shamai. Linear Precoding via Conic Optimization for Fixed MIMO Receivers. *IEEE Transactions on Signal Processing*, 54(1):161–176, 2006.

[6] Sergiy A Vorobyov, Alex B Gershman, and Zhi-Quan Luo. Robust Adaptive Beamforming Using Worst-Case Performance Optimization: a Solution to the Signal Mismatch Problem. *IEEE Transactions on Signal Processing*, 51(2):313–324, 2003.

[7] Ioannis D Schizas, Alejandro Ribeiro, and Georgios B Giannakis. Consensus in Ad Hoc WSNs with Noisy Links—Part I: Distributed Estimation of Deterministic Signals. *IEEE Transactions on Signal Processing*, 56(1):350–364, 2008.

[8] Giuseppe Bianchi. Performance Analysis of the IEEE 802.11 Distributed Coordination Function. *IEEE Journal on selected areas in communications*, 18(3):535–547, 2000.

[9] Paul Patras, Albert Banchs, Pablo Serrano, and Arturo Azcorra. A Control-Theoretic Approach to Distributed Optimal Configuration of 802.11 WLANs. *IEEE Transactions on Mobile Computing*, 10(6):897–910, 2011.

[10] Vasilios A Siris and George Stamatakis. Optimal CWmin Selection for Achieving Proportional Fairness in Multi-Rate 802.11 e WLANs: Test-bed Implementation and Evaluation. In *Proceedings of the 1st international workshop on Wireless network testbeds, experimental evaluation & characterization*, pages 41–48. ACM, 2006.

[11] Li Bin Jiang and Soung Chang Liew. Proportional Fairness in Wireless LANs and Ad Hoc Networks. In *Wireless Communications and Networking Conference, 2005 IEEE*, volume 3, pages 1551–1556. IEEE, 2005.

[12] Alessandro Checco and Douglas J Leith. Proportional Fairness in 802.11 Wireless LANs. *IEEE Communications Letters*, 15(8):807–809, 2011.

[13] Massimiliano Laddomada, Fabio Mesiti, Marina Mondin, and Fred Daneshgaran. A Novel Proportional Fairness Criterion for Throughput Allocation in Multirate IEEE 802.11. *arXiv preprint arXiv:0809.1061*, 2008.

[14] Cristina Cano and Gergely Neu. Wireless Optimisation via Convex Bandits: Unlicensed LTE/WiFi Coexistence. *arXiv preprint arXiv:1802.04327*, 2018.

[15] Alekh Agarwal, Ofer Dekel, and Lin Xiao. Optimal Algorithms for Online Convex Optimization with Multi-Point Bandit Feedback. In *COLT*, pages 28–40. Citeseer, 2010.

[16] Abraham D Flaxman, Adam Tauman Kalai, and H Brendan McMahan. Online Convex Optimization in the Bandit Setting: Gradient Descent without a Gradient. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 385–394. Society for Industrial and Applied Mathematics, 2005.

[17] Martin Zinkevich. Online Convex Programming and Generalized Infinitesimal Gradient Ascent. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 928–936, 2003.

[18] Ankan Saha and Ambuj Tewari. Improved Regret Guarantees for Online Smooth Convex Optimization with Bandit Feedback. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 636–642, 2011.

[19] Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. Amendment 4: Enhancements for Very High Throughput for Operation in Bands Below 6 GHz. *ANSI/IEEE Standard 802.11ac*, 2013.

[20] Cristina Cano, Boris Bellalta, Anna Sfairopoulou, and Jaume Barceló. Tuning the EDCA Parameters in WLANs with Heterogeneous Traffic: A Flow-Level Analysis. *Computer Networks*, 54(13):2199–2214, 2010.

[21] Pablo Serrano Yáñez-Mingot. Estrategias de Configuración de Redes WLAN IEEE 802.11 e EDCA. 2006.

[22] Vasilios A Siris and Matina Kavouridou. Achieving Service Differentiation and High Utilization in IEEE 802.11. In *IFIP International Conference on Personal Wireless Communications*, pages 128–137. Springer, 2003.

[23] Juliana Freitag, Nelson LS da Fonseca, and José Ferreira de Rezende. Tuning of 802.11 e Network Parameters. *IEEE Communications letters*, 10(8):611–613, 2006.

[24] Boris Bellalta, Cristina Cano, Miquel Oliver, and Michela Meo. Modeling the IEEE 802.11 e EDCA for MAC Parameter Optimization. In *Proc. of the Performance Modelling and Evaluation of Heterogeneous Networks Conference (Het-Nets 06)*, 2006.

[25] Paul Patras, Andrés Garcia-Saavedra, David Malone, and Douglas J Leith. Rigorous and Practical Proportional-Fair Allocation for Multi-Rate Wi-Fi. *Ad Hoc Networks*, 36:21–34, 2016.

[26] Víctor Valls and Douglas J Leith. Proportional Fair MU-MIMO in 802.11 WLANs. *IEEE Wireless Communications Letters*, 3(2):221–224, 2014.